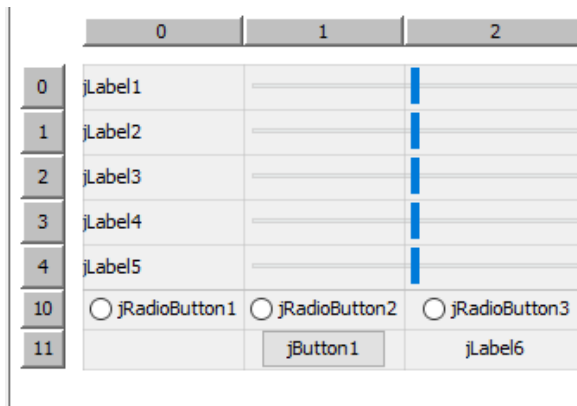


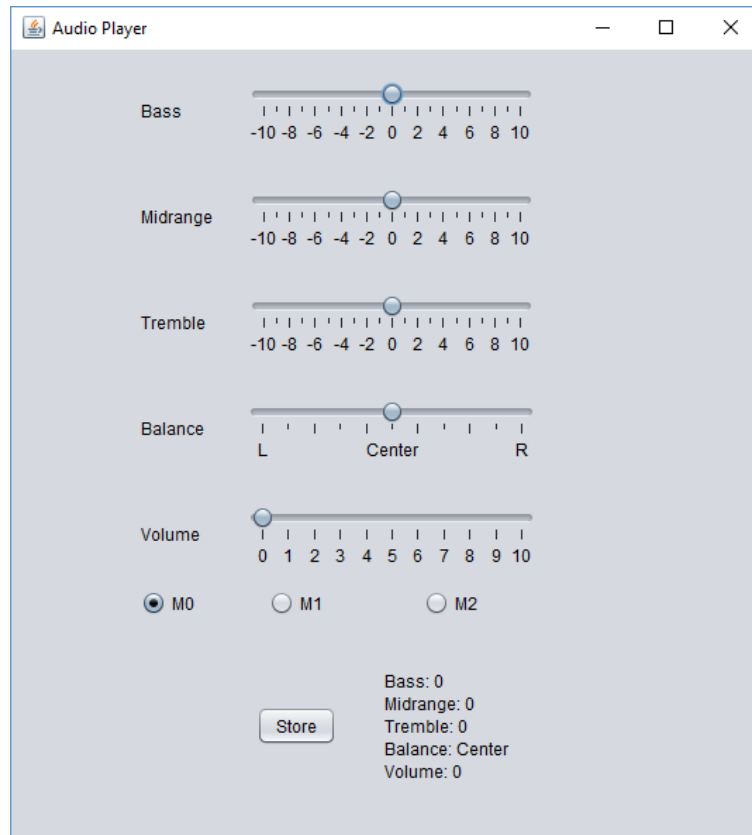
## Εργαστήριο 4 - Άσκηση - Ανάλυση

**Εκφώνηση:** Δημιουργείτε την εφαρμογή «Audio Player» σε Java Swing με χρήση NetBeans ακολουθώντας τις παρακάτω οδηγίες.

1. Η εφαρμογή θα σχεδιασθεί σε ένα εξωτερικό υποδοχέα JFrame, ο οποίος θα έχει
  - τίτλο «Audio Player»,
  - διάταξη τύπου Grid Bag Layout.
2. Τοποθετήστε στο frame
  - 6 ετικέτες
  - 5 slider
  - 3 radio button
  - 1 button
3. Επιλέξτε «Customize Layout» από το μενού του frame που ανοίγει με το δεξί πλήκτρο του ποντικιού. Τοποθετήστε τα συστατικά όπως φαίνεται παρακάτω:  
(Σημείωση με την επιλογή grid size μπορείτε ένα συστατικό να το τοποθετείτε σε περισσότερα του ενός κελιά ενώ με το Anchor τα τοποθετείτε σχετικά στη στήλη, π.χ. αν επιλέξετε το Line Start θα είναι τοποθετημένα αριστερά οριζόντια).



4. Ονομάστε τα συστατικά text και variable name όπως φαίνεται στην παρακάτω εικόνα – για παράδειγμα για το πρώτο ορίστε για το jLabel το text: «Bass» και το variable name: «LBass» ενώ το variable name του jSlider «SlBass». Ονομάστε το variable name του τελευταίου label (jLabel6) «info». Τοποθετείστε το σε δύο κελιά και ορίστε μέγεθος του (preferredSize) 100, 150.



5. Για το slider Bass Midrange, Tremble ορίστε τις παρακάτω ιδιότητες ως εξής:
  1. majorTickSpacing: 2
  2. minorTickSpacing: 1
  3. minimum: -10
  4. maximum: +10
  5. value: 0
  6. paintTicks: YES
  7. paintLabels: YES
6. Για το slider Volume ορίστε τις παρακάτω ιδιότητες ως εξής:
  1. majorTickSpacing: 1
  2. minimum: 0
  3. maximum: +10
  4. value: 0
  5. paintTicks: YES
  6. paintLabels: YES
7. Για το slider Balance ορίστε τις παρακάτω ιδιότητες ως εξής:
  1. majorTickSpacing: 2
  2. minorTickSpacing: 1

3. minimum: -5
4. maximum: +5
5. value: 0
6. paintTicks: YES
7. paintLabels: YES
8. Ανοίξτε το μενού του slider 4 (Balance) με το δεξί πλήκτρο του ποντικιού και επιλέξτε «customize Code» και προσθέστε στο κατάλληλο σημείο τον παρακάτω κώδικα:

```
table.put(new Integer(0), new JLabel("Center"));
table.put(new Integer(-5), new JLabel("L"));
table.put(new Integer(5), new JLabel("R"));
SlBalance.setLabelTable(table);
```

Σημείωση: μην ξεχάσετε να δηλώσετε στις γενικές σας μεταβλητές τον πίνακα table ως

```
Hashtable table = new Hashtable();
```

Θα *χρειαστεί* να προσθέσετε δύο βιβλιοθήκες - εκμεταλλευτείτε τις δυνατότητες του netbeans.

Τρέξτε το πρόγραμμα για να μελετήσετε τι κάνει το παραπάνω κομμάτι κώδικα.

9. Προσπαθήστε να κάνετε εύχρηστη την διεπαφή σας χρησιμοποιώντας για τα sliders το empty border. Πειραματιστείτε.
10. Δημιουργήστε μία βοηθητική τάξη Presets με τον κατάλληλο δομητή η οποία θα αποθηκεύει τιμές για τα πέντε sliders.
11. Δηλώστε ένα πίνακα Presets.
12. Δημιουργήστε μία μέθοδο **setupPresets** η οποία θα δίνει τις παρακάτω αρχικές τιμές στα πέντε slider ανά radiobutton:
  - a. M0 (0,0,0,0,0)                      Presets[0]
  - b. M1 (1, -1, 9, 0, 4)                  Presets[1]
  - c. M2 (2, 4, -2, 4, 2)                  Presets[2]

Μην ξεχάσετε να καλέσετε την μέθοδο με την εκκίνηση της εφαρμογής σας αμέσως μετά την initComponents().

13. Δημιουργήστε μία μέθοδο **showSliderValues** η οποία θα εμφανίζει τις τιμές των πέντε slider στην ετικέτα info. (Προτείνεται η χρήση της HTML)
14. Εμφανίστε στην Ετικέτα info την τρέχουσα κατάσταση των πέντε sliders όποτε συμβαίνει μία αλλαγή σε κάποιο συστατικό της διεπιφάνειας (Change - stateChanged).

15. Δημιουργήστε μία μέθοδο **loadPresets** φόρτωσης των αποθηκευμένων τιμών όποτε πατείτε το αντίστοιχο radio button (προγραμματίστε και το αντίστοιχο γεγονός).
16. Δημιουργήστε μία μέθοδο **storePresets** αποθήκευσης και προγραμματίστε το κουμπί JButton ώστε να αποθηκεύει τις τρέχουσες τιμές στα αντίστοιχα radiobutton.

Σημείωση: Μέθοδοι που πιθανώς θα σας φανούν χρήσιμες:

- JTextField.setText(string);
- JTextField.getText();
- Void JSlider.setValue(int)
- Int JSlider.getValue()

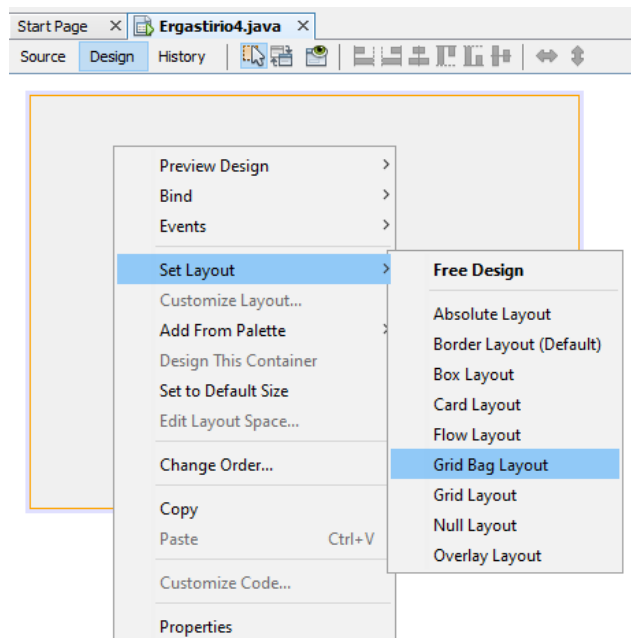
### Ανάλυση και εξήγηση λύσης

**Βήμα 1:** Δημιουργούμε ένα νέο project στο NetBeans με όνομα Askisi6.

**Βήμα 2:** Δημιουργούμε ένα νέο Java αρχείο τύπου JFrame Form με όνομα Ergastirio4.

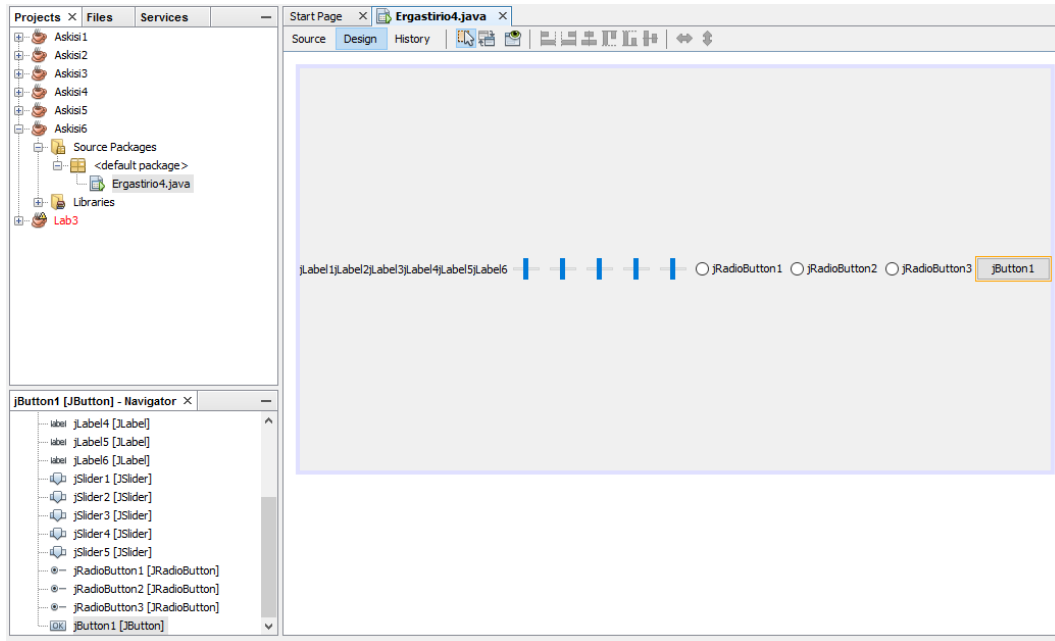
**Βήμα 3:** Ορισμός Ιδιοτήτων στο JFrame. Ορίζουμε ως τίτλο του JFrame το «Audio Player».

**Βήμα 4:** Ορίζουμε τη διάταξη σε Grid Bag Layout (Εικόνα 1).



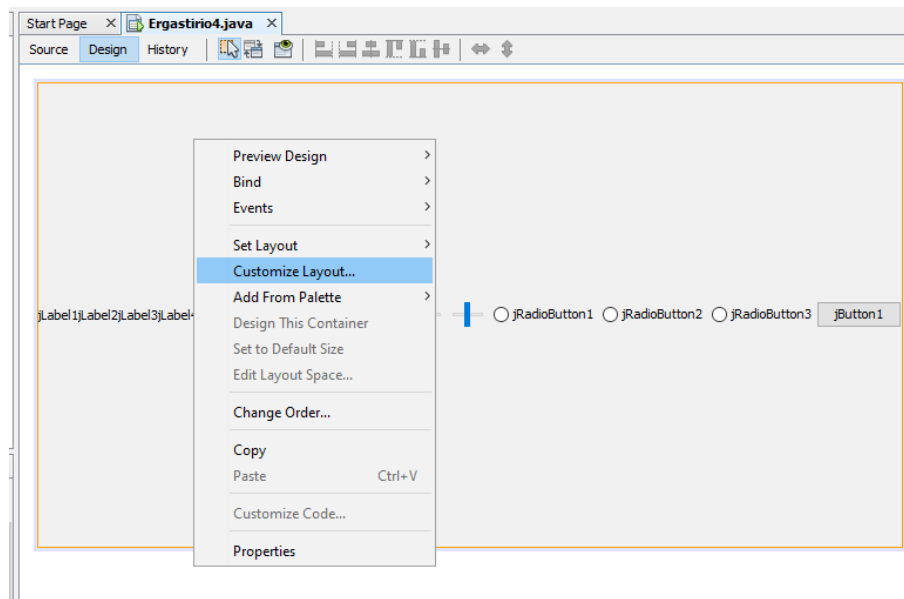
Εικόνα 1: Ορισμός της διάταξης

**Βήμα 5:** Τοποθετούμε στο JFrame έξι ετικέτες, πέντε slider, τρία radio button και ένα jButton, όπως φαίνεται στην Εικόνα 2.

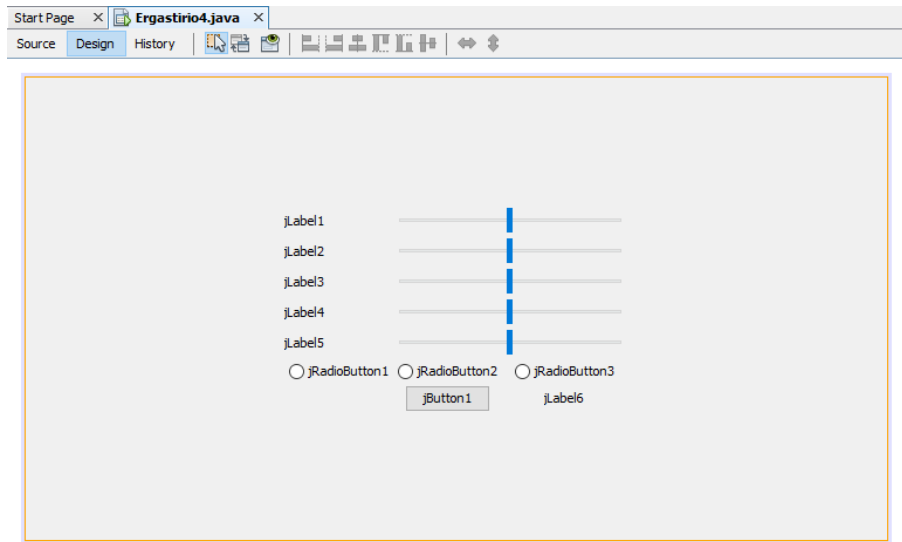


Εικόνα 2: Τοποθέτηση συστατικών στο JFrame

**Βήμα 6:** Με χρήση του εργαλείου Customize Layout (δεξί click στον καμβά για να επιλέξω το Customize Layout - Εικόνα 3) τοποθετώ τα συστατικά όπως μας ζητάει η εκφώνηση (Εικόνα 4).

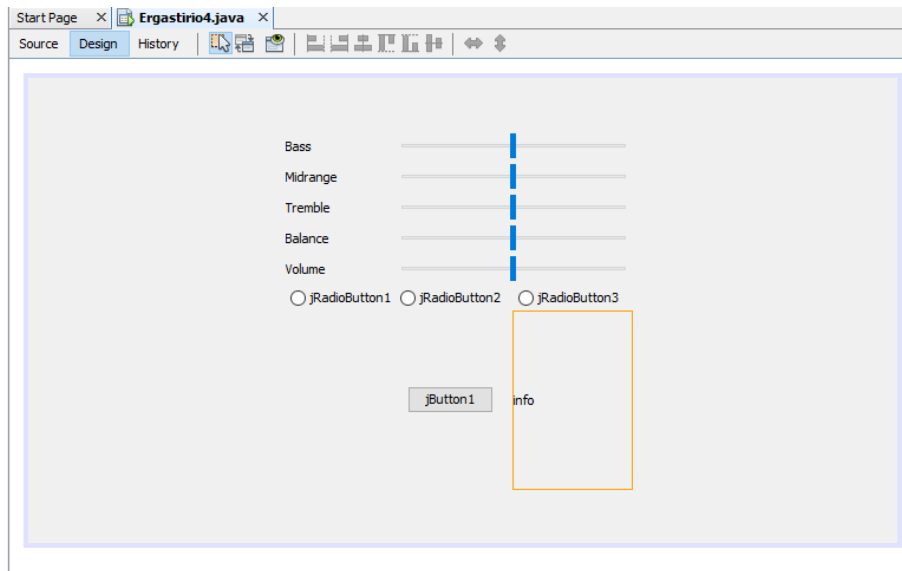


Εικόνα 3: Customize Layout



Εικόνα 4: Η εμφάνιση των συστατικών μετά τη χρήση του Customize Layout

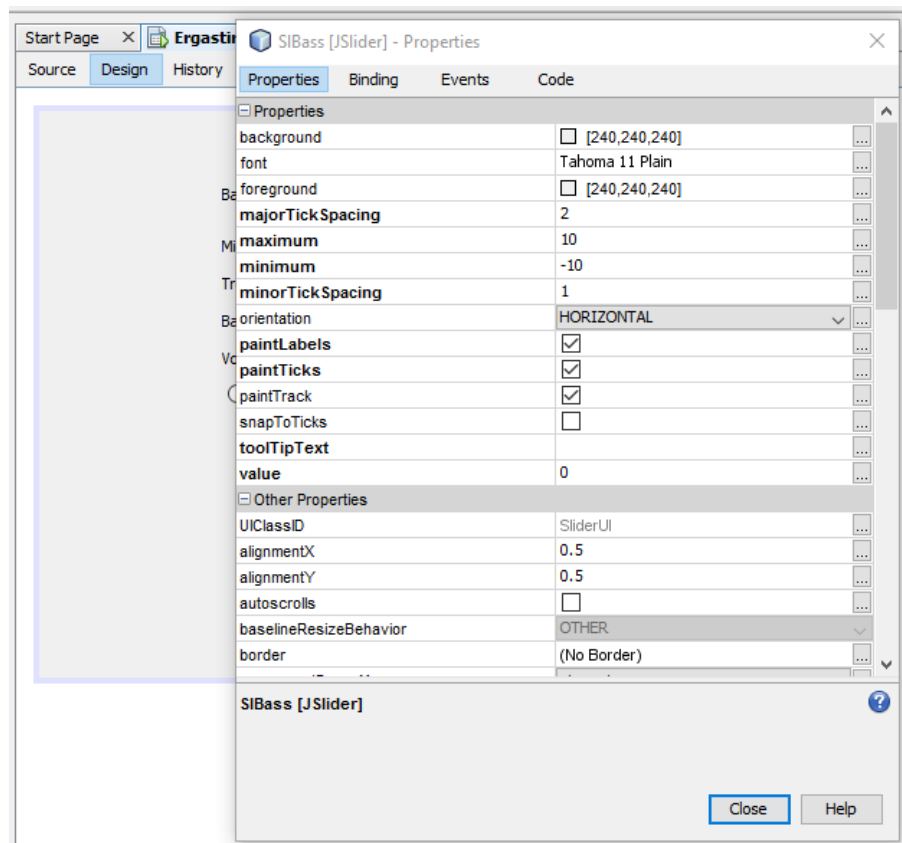
**Βήμα 7:** Αλλάζουμε text και variable names όπως αναφέρει η εκφώνηση και το αποτέλεσμα φαίνεται στην Εικόνα 5.



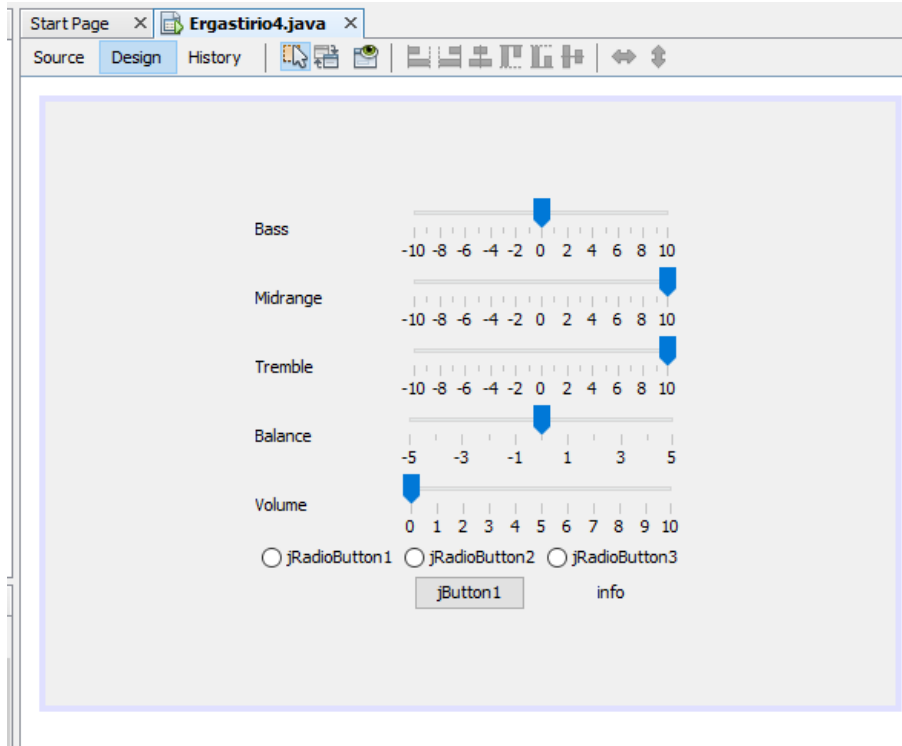
Εικόνα 5: Η εμφάνιση των συστατικών μετά την αλλαγή text των συστατικών

**Βήμα 8:** Για τα πέντε slider ορίζουμε επτά ιδιότητες, όπως αναφέρονται στην εκφώνηση. Για παράδειγμα στην Εικόνα 6 φαίνεται ο ορισμός των ιδιοτήτων για το πρώτο slider. Οι ιδιότητες που ορίζουμε είναι: (α) majorTickSpacing, κάθε πόσες θέσεις θα εμφανίζεται η μεγάλη γραμμή κάτω από το slider, (β) minorTickSpacing, κάθε πόσες θέσεις θα εμφανίζεται η μικρή γραμμή κάτω από το slider, (γ) minimum, κάτω άκρο του slider, (δ) maximum, πάνω άκρο του slider, (ε) value, αρχική τιμή που θα εμφανίζεται ο δείκτης του slider, (στ) paintTicks, ορίζει αν θα εμφανιστούν οι γραμμές κάτω από το slider και (ζ) paintLabels, ορίζει αν θα εμφανιστούν οι

ετικέτες (αριθμοί) κάτω από το slider. Στην Εικόνα 7 παρουσιάζεται η εμφάνιση της εφαρμογής μετά τον ορισμό των παραπάνω ιδιοτήτων στα πέντε slider.



Εικόνα 6: Ορισμός ιδιοτήτων του SIBass



Εικόνα 7: Η εμφάνιση των συστατικών μετά τον ορισμό των ιδιοτήτων στα slider

**Βήμα 9:** Στο 4<sup>ο</sup> slider, το slider balance θέλουμε οι ετικέτες να είναι διαφορετικές από το standard set ετικετών, δηλαδή τους αριθμούς. Θέλουμε στο αριστερό άκρο του slider να μπει η ετικέτα Left (στη θέση του -5), στο δεξί άκρο να μπει η ετικέτα Right (στη θέση του 5) και στο κέντρο το Center (στη θέση του 0).

Αυτό θα το υλοποιήσουμε με χρήση ενός Hashtable. Έτσι, δηλώνουμε ως global μεταβλητή ένα Hashtable, όπως βλέπουμε στην Εικόνα 8. Επίσης, πρέπει να προσθέσουμε μία βιβλιοθήκη για το Hashtable, την οποία μπορούμε να την προσθέσουμε με χρήση της επιλογής *Fix Imports*, του pop up μενού που ανοίγει με δεξί click πάνω στο Hashtable.

Τις ετικέτες θα τις ορίσουμε στο design, στον κώδικα υλοποίησης του slider. Έτσι, με δεξί click πάνω στο slider balance, επιλέγουμε από το pop up μενού που ανοίγει, την επιλογή *Customize Code*, όπως φαίνεται στην Εικόνα 9. Εκεί προσθέτουμε τον κατάλληλο κώδικα, όπως φαίνεται στην Εικόνα 10. Στις παρακάτω γραμμές κώδικα,

```
table.put(new Integer(0), new JLabel("Center"));
table.put(new Integer(-5), new JLabel("L"));
table.put(new Integer(5), new JLabel("R"));
```

στο Hashtable πίνακα *table*, ουσιαστικά ορίζουμε ότι στη θέση 0 του slider θα μπει η ετικέτα "Center", στη θέση -5 η ετικέτα "L" και στη θέση 5 η ετικέτα "R". Ενώ στη γραμμή κώδικα

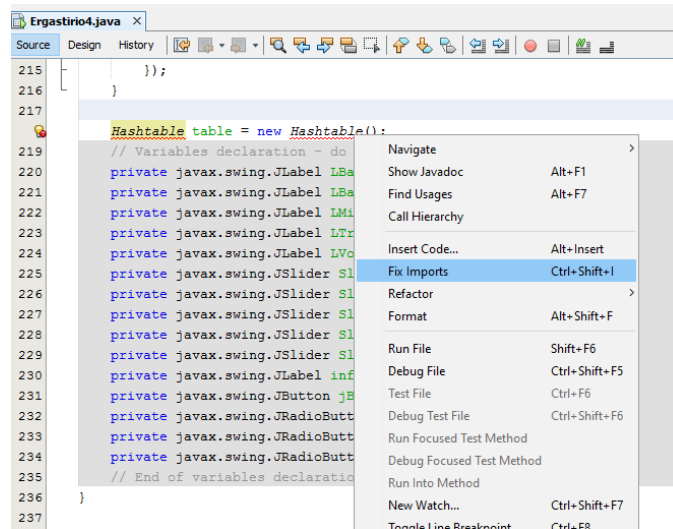


SLBalance.setLabelTable(table);

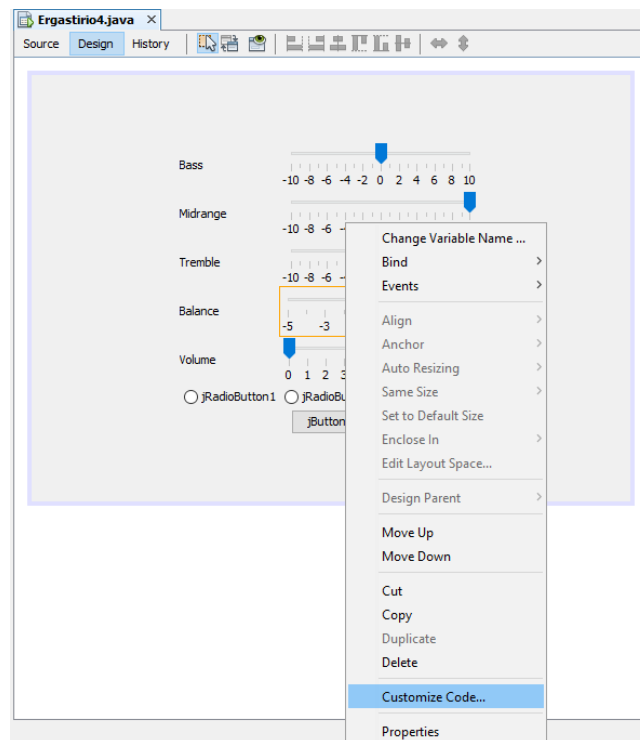
Ορίζουμε ότι οι ετικέτες του slider SLBalance θα οριστούν με βάση το *table*.

Για να ολοκληρώσουμε αυτό το βήμα θα πρέπει να προσθέσουμε μία βιβλιοθήκη για χρήση της JLabel, με χρήση της γραμμής κώδικα `import javax.swing.JLabel;`

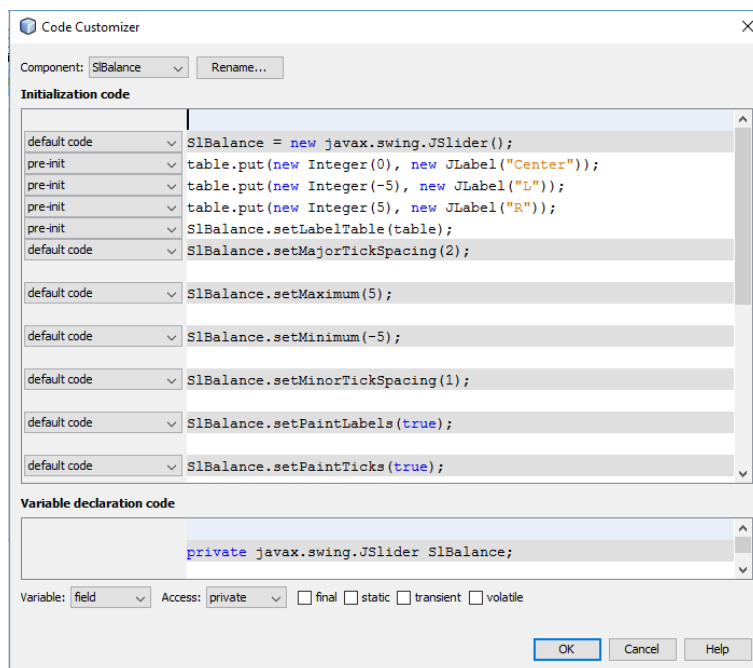
Οι νέες ετικέτες θα εμφανιστούν στην εκτέλεση του προγράμματος.



Εικόνα 8: Δήλωση Hashtable και χρήση της Fix Imports



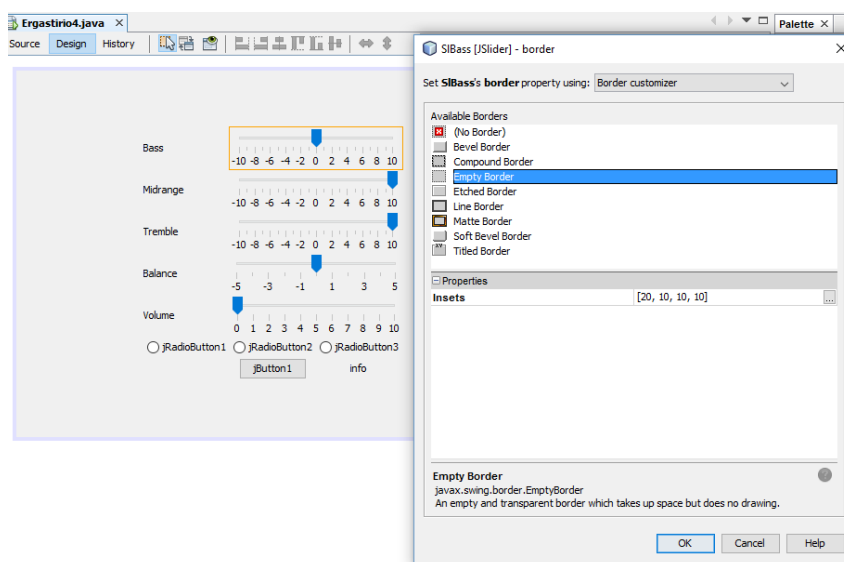
Εικόνα 9: Χρήση του Customize Code



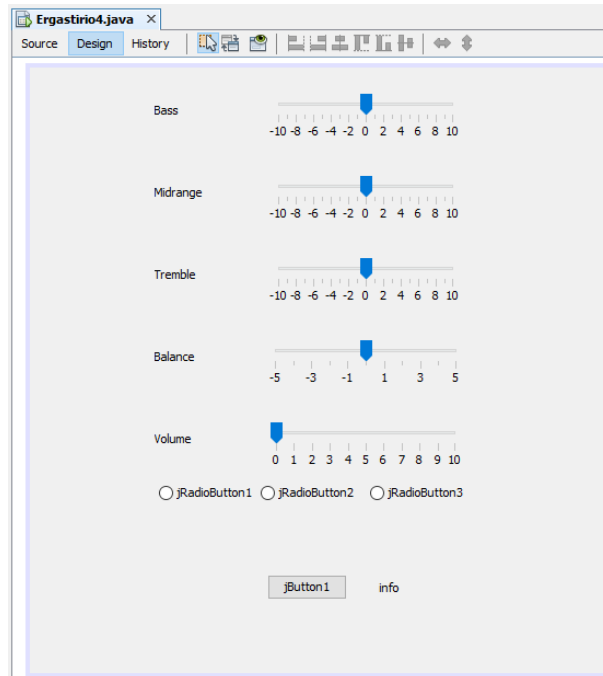
Εικόνα 10: Ο κώδικας του Customize Code στο slider balance

**Βήμα 10:** Θα βελτιώσουμε τώρα την εμφάνιση της εφαρμογής μας. Αυτό θα το πετύχουμε απομακρύνοντας τα slider για να γίνουν πιο ευδιάκριτα με χρήση των empty borders. Έτσι, επιλέγουμε για κάθε slider την ιδιότητα border και επιλέγουμε το τύπο empty border. Ορίζουμε τις τιμές 20, 10, 10 και 10, όπως φαίνεται στην Εικόνα 11.

Επίσης, ορίζουμε preferredSize και minimumSize του JFrame τις τιμές 550 και 600 για να μην «συμπιεστούν» τα συστατικά από ένα μικρό σε μέγεθος Frame. Στην Εικόνα 12, βλέπουμε την εμφάνιση του design μετά την εφαρμογή των παραπάνω.



Εικόνα 11: Χρήση του empty border για τα πέντε slider



Εικόνα 12: Η εμφάνιση του design μετά το βήμα 10

**Βήμα 11:** Για να διαχειριστούμε τα radiobuttons ως μνήμες στο audioplayer, θα πρέπει να αποθηκεύουμε τρία σέτ τιμών των sliders σε μία δομή. Αυτό θα το κάνουμε με τη δημιουργία της τάξης Presets. Έτσι, θα δημιουργήσουμε ένα αρχείο Java Class όπου εκεί θα ορίσουμε μία κλάση Presets με πέντε ακέραιες ιδιότητες, μία για κάθε δείκτη των πέντε slider – ένας δείκτης στα slider δείχνει μία ακέραια τιμή. Στην Εικόνα 13 φαίνεται η κλάση Presets. Επίσης, δηλώνουμε ένα πίνακα αντικειμένων Presets με διάσταση τρία, όσα είναι τα radiobuttons με την εντολή `Presets pinakas[] = new Presets[3];`

```

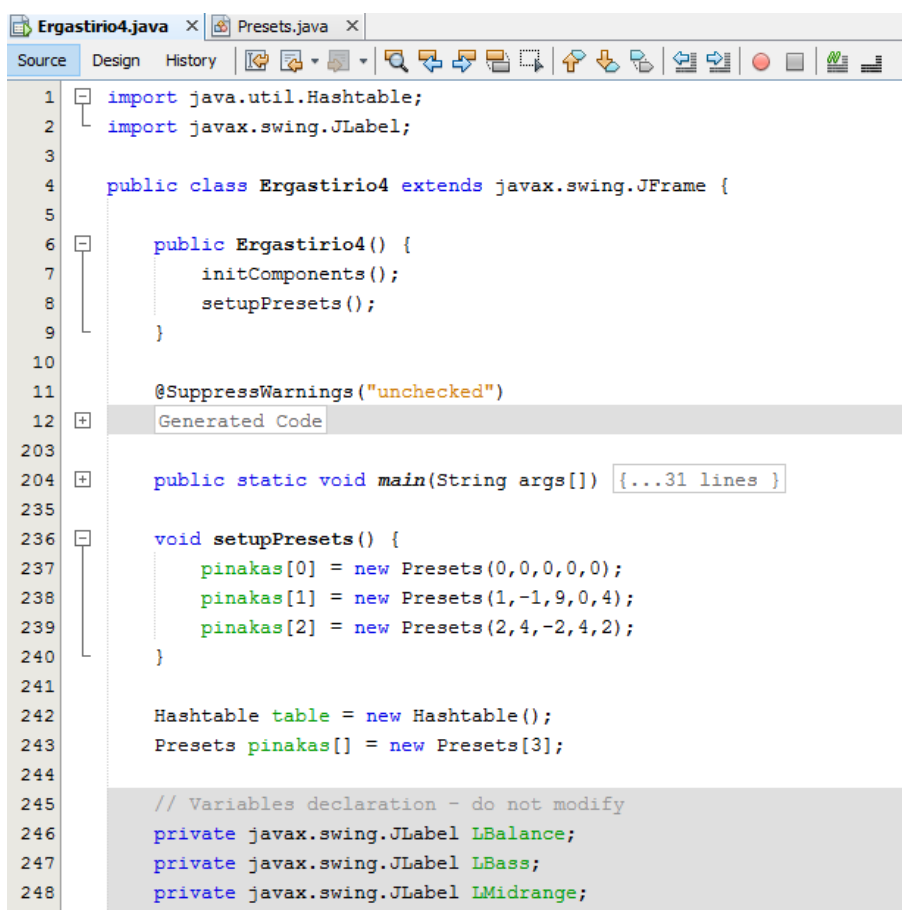
1  public class Presets {
2      int bass;
3      int midrange;
4      int tremble;
5      int balance;
6      int volume;
7
8      Presets(int a, int b, int c, int d, int e) {
9          bass = a;
10         midrange = b;
11         tremble = c;
12         balance = d;
13         volume = e;
14     }
15 }

```

Εικόνα 13: Η τάξη Presets

**Βήμα 12:** Σε αυτό το βήμα θα δώσουμε αρχικές τιμές στα τρία αντικείμενα του πίνακα. Πριν το κάνουμε αυτό θα διαμορφώσουμε την εμφάνιση των τριών radiobutton. Το κείμενο των τριών radiobuttons θα γίνει M0, M1 και M2 και οι μεταβλητές τους RBM0, RBM1, και RBM2 αντίστοιχα. Φροντίζουμε να «μπουν» και τα τρία στο ίδιο buttongroup και στο RBM0 να οριστεί η ιδιότητα selected ώστε να εμφανίζεται επιλεγμένο το πρώτο radiobutton όταν ξεκινά η εφαρμογή. Επίσης, αλλάζουμε, από το *customize layout* το insets των στοιχείων (radiobutton και jbutton) για καλύτερη στοίχιση και εμφάνιση.

Για την απόδοση αρχικών τιμών στα τρία αντικείμενα του πίνακα δημιουργούμε τη μέθοδο `setupPresets` - Εικόνα 14, γραμμές κώδικα 236 - 240. Την μέθοδο την καλούμε από το δομητή της τάξης μετά την `initComponents()` για να αποδοθούν οι αρχικές τιμές στην εκκίνηση της εφαρμογής.



```

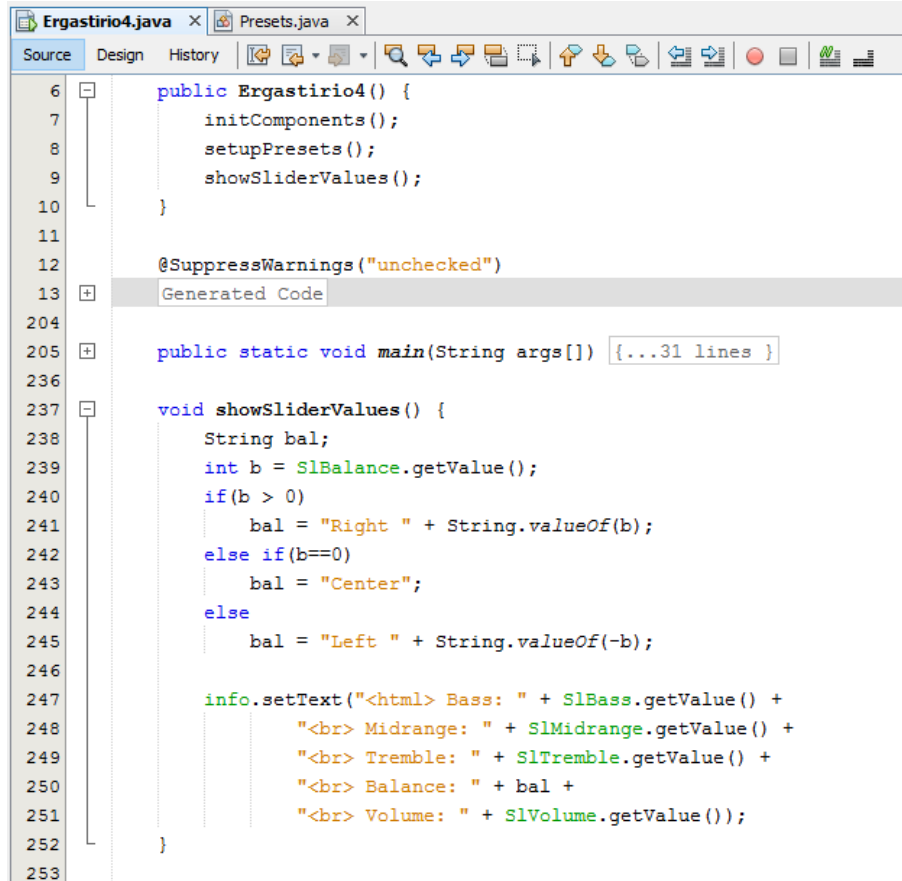
1  import java.util.Hashtable;
2  import javax.swing.JLabel;
3
4  public class Ergastirio4 extends javax.swing.JFrame {
5
6      public Ergastirio4() {
7          initComponents();
8          setupPresets();
9      }
10
11      @SuppressWarnings("unchecked")
12      Generated Code
13
14      public static void main(String args[]) { ...31 lines }
15
16      void setupPresets() {
17          pinakas[0] = new Presets(0,0,0,0,0);
18          pinakas[1] = new Presets(1,-1,9,0,4);
19          pinakas[2] = new Presets(2,4,-2,4,2);
20      }
21
22      Hashtable table = new Hashtable();
23      Presets pinakas[] = new Presets[3];
24
25      // Variables declaration - do not modify
26      private javax.swing.JLabel LBalance;
27      private javax.swing.JLabel LBass;
28      private javax.swing.JLabel LMidrange;

```

Εικόνα 14: Η μέθοδος `setupPresets`

**Βήμα 13:** Το επόμενο που θα υλοποιήσουμε είναι η εμφάνιση της θέσης των δεικτών των slider στην ετικέτα info. Για να εμφανίσουμε τις τιμές των πέντε δεικτών σε πέντε σειρές θα χρησιμοποιήσουμε html. Επίσης, τη θέση του δείκτη του slider balance θα την παρουσιάσουμε σε συνάρτηση με τις ετικέτες Left, Center και Right. Στην Εικόνα 15, φαίνεται ο κώδικας υλοποίησης της μεθόδου `showSliderValues`. Με τη μέθοδο `getValue` (π.χ. γραμμή κώδικα 239)

παίρνουμε την ακέραια τιμή που δείχνει ο δείκτης του slider που δείχνει. Για να γράψουμε html σε μία ετικέτα, θα πρέπει στη `setText` να ξεκινήσουμε το κείμενο με `<html>` (γραμμή κώδικα 247). Για να αλλάξουμε σειρά χρησιμοποιούμε το tag της html το `<br>` (π.χ. γραμμή κώδικα 248). Τέλος, την μέθοδο `showSliderValues` την καλούμε από το δομητή της τάξης μετά την `initComponents()` (π.χ. γραμμή κώδικα 9) για να εμφανιστούν οι αρχικές τιμές στην ετικέτα `info` στην εκκίνηση της εφαρμογής.



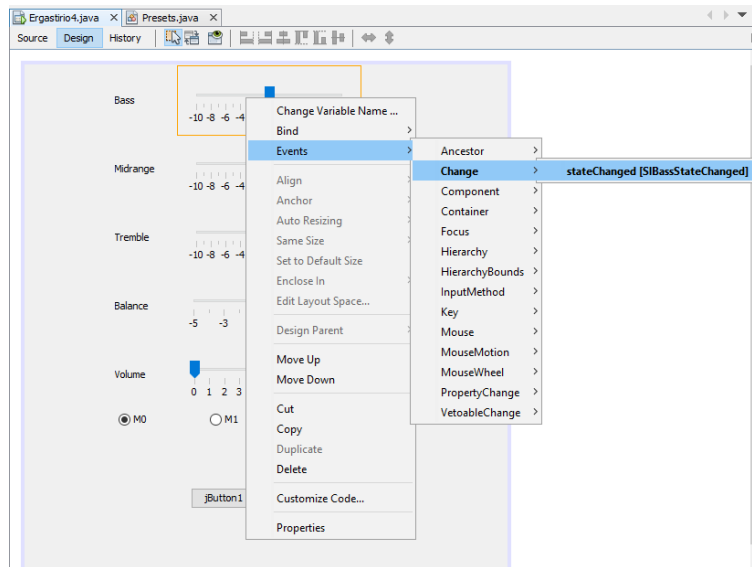
```

6      public Ergastirio4() {
7          initComponents();
8          setupPresets();
9          showSliderValues();
10     }
11
12     @SuppressWarnings("unchecked")
13     Generated Code
14
204
205     public static void main(String args[]) { ...31 lines }
206
236
237     void showSliderValues() {
238         String bal;
239         int b = SlBalance.getValue();
240         if(b > 0)
241             bal = "Right " + String.valueOf(b);
242         else if(b==0)
243             bal = "Center";
244         else
245             bal = "Left " + String.valueOf(-b);
246
247         info.setText("<html> Bass: " + SlBass.getValue() +
248             "<br> Midrange: " + SlMidrange.getValue() +
249             "<br> Tremble: " + SlTremble.getValue() +
250             "<br> Balance: " + bal +
251             "<br> Volume: " + SlVolume.getValue());
252     }
253

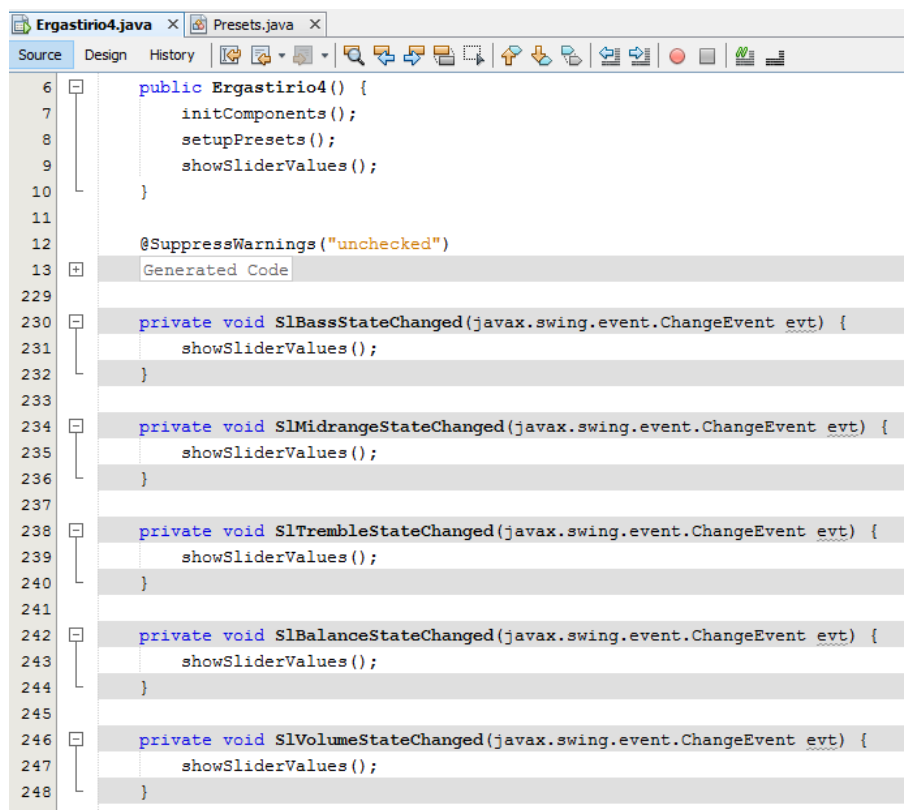
```

Εικόνα 15: Η μέθοδος `showSliderValues`

**Βήμα 14:** Στόχος της εφαρμογής είναι να εμφανίζει άμεσα την τρέχουσα τιμή κάθε δείκτη των slider. Αυτό θα το υλοποιήσουμε με τη δημιουργία του γεγονότος `State-Changed`, όπως φαίνεται στην Εικόνα 16. Στην ενέργεια κάθε γεγονότος θα γίνει κλήση της `showSliderValues`.



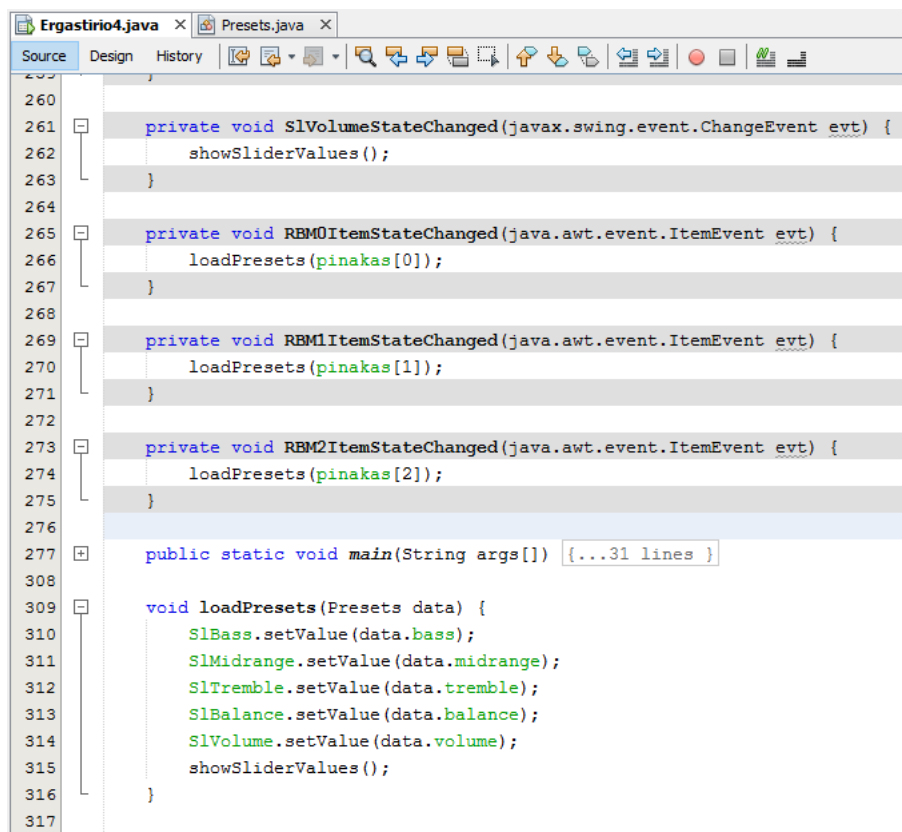
Εικόνα 16: Δημιουργία του γεγονότος State-Changed ενός slider



Εικόνα 17: Ο κώδικας των γεγονότων State-Changed των sliders

**Βήμα 15:** Σε αυτό το βήμα θα υλοποιήσουμε τη χρήση των τριών radiobutton μνημών. Έτσι, κάθε φορά που θα πατάμε ένα radiobutton θα πρέπει να αλλάζουν οι θέσεις των δεικτών των slider σύμφωνα με τις αποθηκευμένες τιμές στο αντίστοιχο αντικείμενο του πίνακα της τάξης Presets. Ταυτόχρονα θα ενημερώνουμε κατάλληλα την εμφάνιση των τιμών των δεικτών στην

ετικέτα info, καλώντας τη μέθοδο showSliderValues. Αυτά θα τα υλοποιήσουμε δημιουργώντας τη μέθοδο loadPresets (γραμμές κώδικα 309-316). Με τη μέθοδο setValue() τοποθετούμε το δείκτη ενός slider στη θέση που δείχνει ο ακέραιος αριθμός που περιέχει η παράμετρος της μεθόδου (π.χ. data.bass). Η κλήση της loadPresets, γίνεται από το κατάλληλο γεγονός itemStateChanged των τριών radioButtons (γραμμές κώδικα 265-275).



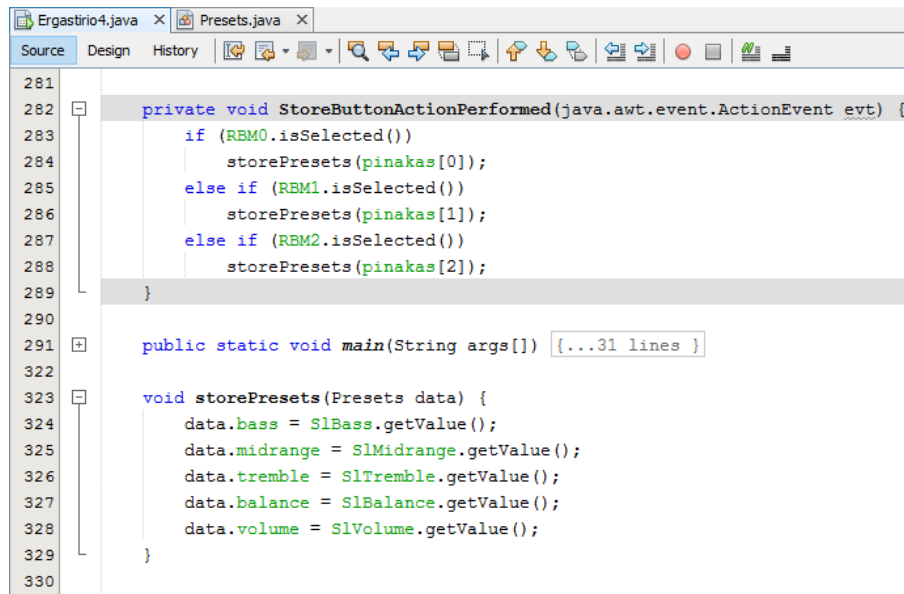
```

260
261 private void SlVolumeStateChanged(javax.swing.event.ChangeEvent evt) {
262     showSliderValues();
263 }
264
265 private void RBM0ItemStateChanged(java.awt.event.ItemEvent evt) {
266     loadPresets(pinakas[0]);
267 }
268
269 private void RBM1ItemStateChanged(java.awt.event.ItemEvent evt) {
270     loadPresets(pinakas[1]);
271 }
272
273 private void RBM2ItemStateChanged(java.awt.event.ItemEvent evt) {
274     loadPresets(pinakas[2]);
275 }
276
277 public static void main(String args[]) { ...31 lines }
278
279
308
309 void loadPresets(Presets data) {
310     SlBass.setValue(data.bass);
311     SlMidrange.setValue(data.midrange);
312     SlTremble.setValue(data.tremble);
313     SlBalance.setValue(data.balance);
314     SlVolume.setValue(data.volume);
315     showSliderValues();
316 }
317

```

Εικόνα 18: Η μέθοδος loadPresets και τα γεγονότα των radiobutton κλήσης της

**Βήμα 16:** Τελευταίο βήμα για την ολοκλήρωση της εφαρμογής είναι ο προγραμματισμός της αλλαγής των τιμών μίας μνήμης. Για την αποθήκευση των νέων τιμών των δεικτών των slider στο κατάλληλο αντικείμενο του πίνακα χρησιμοποιείται η μέθοδος storePresets (γραμμές κώδικα 323-329). Η κλήση της storePresets, γίνεται από το γεγονός ActionPerformed του Store jButton (γραμμές κώδικα 282-289) με παράμετρο το αντικείμενο του πίνακα Presets που αντιστοιχεί στο radiobutton που είναι επιλεγμένο την στιγμή που πατήθηκε το Store jButton.



```
281
282 private void StoreButtonActionPerformed(java.awt.event.ActionEvent evt) {
283     if (RBM0.isSelected())
284         storePresets(pinakas[0]);
285     else if (RBM1.isSelected())
286         storePresets(pinakas[1]);
287     else if (RBM2.isSelected())
288         storePresets(pinakas[2]);
289 }
290
291 public static void main(String args[]) { ...31 lines }
292
293 void storePresets(Presets data) {
294     data.bass = SlBass.getValue();
295     data.midrange = SlMidrange.getValue();
296     data.tremble = SlTremble.getValue();
297     data.balance = SlBalance.getValue();
298     data.volume = SlVolume.getValue();
299 }
300
301
```

Εικόνα 19: Η μέθοδος storePresets και το γεγονός του button store